UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/625,812	07/26/2000	Timothy J. Van Hook	00100.00.0105	8263

29153 7590 07/10/2009

ADVANCED MICRO DEVICES, INC.
C/O VEDDER PRICE P.C.
222 N.LASALLE STREET
CHICAGO, IL 60601

EXAMINER	
HSU, JONI	

ART UNIT	PAPER NUMBER
2628	

MAIL DATE	DELIVERY MODE
07/10/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/625,812	VAN HOOK, TIMOTHY J.	
	Examiner	Art Unit	
	JONI HSU	2628	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *22 April 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18,23,25-31 and 33* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☒ Claim(s) *25-31* is/are allowed.

6)☒ Claim(s) *1-18,23 and 33* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Response to Arguments

1. Applicant's arguments, see p. 9, filed April 22, 2009, with respect to the 35 U.S.C. 101 rejection has been fully considered and are persuasive. The 35 U.S.C. 101 rejection of Claim 33 has been withdrawn.

2. Applicant's arguments filed April 22, 2009, with respect to 35 U.S.C. 102 (b) and 35 U.S.C. 103 (a) rejections have been fully considered but they are not persuasive.

3. As per Claim 1, Applicant argues that the Davis (US005357617A) programmable processor includes a three stage processor namely a fetch stage, a decode stage and a single "execution stage". Therefore, Davis does not teach a plurality of execution stages and a depth (p. 9).

In reply, the Examiner points out that an "execution stage" is being interpreted as a stage that executes an instruction. Applicant's disclosure describes "the source **fetch** of program 1 (SF1) is **executed**" (p. 14). Therefore, the fetch stage of Davis is considered to be an "execution stage" since it executes the fetch. Similarly, the decoder stage of Davis is also considered to be an "execution stage" since it executes the decoding. Therefore, Davis teaches three execution stages, namely a fetch execution stage, a decode execution stage, and decoded instruction execution stage (col. 2, lines 12-28), and therefore teaches a plurality of execution stages and a depth of three execution stages.

4. As per Claim 33, Applicant argues that Joffe (US006330584B1) teaches that when an instruction is suspended, it is re-executed. This is done for a single task. Joffe does not look to see if all of the plurality of programs are completed (p. 11).

In reply, Examiner points out Joffe describes "If a task attempts to access an unavailable resource, the task is suspended…When the resource becomes available, the suspended task is resumed, and the instruction accessing the resource is re-executed...the task does not get access to the same resource until after **every** other task sharing the resource has **finished** accessing the resource" (col. 2, lines 29-39). If Wait signal is asserted, instruction execution is not completed and PC register is frozen, but task remains active, and instruction will be executed again starting next clock cycle. If Suspend/Wait signals are deasserted, PC register is changed to point to next instruction (col. 10, lines 19-32). So, Joffe teaches checking to see if Suspend/Wait signals are desasserted, which indicates instruction execution is completed (col. 10, lines 19-32) and resource has become available (col. 2, lines 32-34), and resource becomes available after **every** other task sharing resource has **finished** accessing resource (col. 2, lines 29-39). Since the resource only becomes available after **every** other task sharing the resources have **finished** accessing the resource, this means that Joffe teaches checking to see if all of the programs are completed.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6. Claims 1, 2, 4, 7, 9, and 10 are rejected under 35 U.S.C. 102(b) as being anticipated by Davis (US005357617A).

7. As per Claim 1, Davis teaches a programmable processor for executing a plurality of programs, said programmable processor comprising: an executing pipeline having a fetch stage, a decode stage, and an execution stage (col. 2, lines 12-28), and therefore has a depth of a plurality of execution stages (three execution stages). An "execution stage" is being interpreted as a stage that executes an instruction. Applicant's disclosure describes "the source **fetch** of program 1 (SF1) is **executed**" (p. 14). Therefore, the fetch stage of Davis is considered to be an "execution stage" since it executes the fetch. Similarly, the decoder stage of Davis is also considered to be an "execution stage" since it executes the decoding. Therefore, Davis teaches three execution stages, namely a fetch execution stage, a decode execution stage, and decoded instruction execution stage (col. 2, lines 12-28), and therefore teaches a plurality of execution stages and a depth of three execution stages. Therefore there are three or more programs (col. 4, lines 36-39), and therefore a depth (three execution stages) (col. 2, lines 12-28) is less than or equal to said plurality of programs (three or more programs) (col. 4, lines 36-39) wherein each of the plurality of programs comprises a plurality of instructions (col. 2, lines 15-18) and having an average pipeline latency of one instruction per cycle (col. 1, lines 28-30; col. 7, lines 30-34); and an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline (col. 4, lines 13-15; col. 5, lines 52-54; col. 2, lines 12-28; col. 4, lines 36-39).

8. As per Claim 2, Davis teaches wherein said pipeline has a datapath with a depth (three execution stages) equal to said number of programs (three programs) (col. 2, lines 12-28; col. 4, lines 15-18).

9. As per Claim 4, Davis teaches wherein each program of said plurality of programs is independent of the other of said plurality of programs (col. 4, lines 42-45; col. 7, lines 23-26).

10. As per Claim 7, Davis teaches wherein one or more of control and computing resources, instructions, instruction memory, data paths, data memory, and caches are shared by said plurality of programs (col. 2, lines 15-18).

11. As per Claim 9, Davis teaches a single instruction fetch unit for fetching instructions from the instruction memory (col. 2, lines 18-21), and there is inherently an instruction for performing this fetch. Therefore, Davis teaches wherein said instructions comprise load instructions for loading data from a data memory (col. 2, lines 18-21).

12. As per Claim 10, Davis teaches that the executing step further includes storing the working results of the executed instruction in a memory (col. 2, lines 62-64), and there is inherently an instruction for performing this executing step of storing. Therefore, Davis teaches wherein said instructions comprise store instructions for storing data in a memory (col. 2, lines 62-64).

13. Thus, it reasonably appears that Davis describes or discloses every element of Claims 1, 2, 4, 7, 9, and 10 and therefore anticipates the claims subject.

Claim Rejections - 35 USC § 103

14. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

15. The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

16. Claims 3, 5, 6, and 11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) in view of Eickemeyer (US006061710A).

17. As per Claim 3, Davis is relied upon for teachings as discussed above relative to Claim 1.

However, Davis does not teach wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and in the meantime an instruction from another program is being executed by said pipeline. However, Eickemeyer teaches wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed (col. 3, lines 41-48) and in the meantime an instruction from another program is being executed by said pipeline (col. 4, lines 14-25).

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis so that a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and in the meantime an instruction from another program is being executed by said pipeline as suggested by Eickemeyer. Eickemeyer suggests that instructions dependent upon the results of a previously dispatched instruction that has not yet completed causes the pipeline to

stall. For instance, instructions dependent on a load/store instruction in which the necessary data is not in the cache cannot be executed until the data becomes available in the cache. Allowing out-of order completion so that an instruction from another program to be executed by said pipeline in the meantime enables the pipeline to do useful work when a pipeline stall condition is detected instead of being idle and not accomplishing any work while waiting (col. 3, lines 37-63; col. 4, lines 14-25).

18. As per Claim 5, Davis does not teach further including an output buffer for storing out of order data output. However, Eickemeyer teaches further including an output buffer for storing out of order data output (col. 8, line 43-col. 9, line 6).

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis to include an output buffer for storing out of order data output because Eickemeyer suggests that an output buffer is needed in order to reorder the instructions so that they are out of order (col. 8, line 43-col. 9, line 6), and it is advantageous to output data out of order for the same reasons given in the rejection for Claim 3.

19. As per Claim 6, Davis does not teach further including one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs. However, Eickemeyer teaches further including one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs (col. 8, lines 57-63).

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis to include one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs because Eickemeyer

suggests that that a register copy and a program counter are needed in order to ensure that the thread is executing the correct or desired branch path (col. 8, line 43-col. 9, line 6).

20. As per Claim 11, Davis does not teach wherein said data memory comprises a cache. However, Eickemeyer teaches wherein said data memory comprises a cache (col. 10, lines 24-26).

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis so that said data memory comprises a cache because Eickemeyer suggests that cache memories store frequently used and other data nearer the processor and allow instruction execution to continue without waiting the full access time of a main memory (col. 3, lines 21-25).

21. Claim 8 is rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) in view of Nguyen (US005961628A).

Davis is relied upon for the teachings as discussed above relative to Claim 1.

However, Davis does not teach wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N. However, Nguyen teaches processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N (col. 1, lines 11-24, 53-60).

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis to include executing in parallel instructions having SIMD vector lengths that sum up to N because Nguyen teaches fast speed for repetitive tasks (col. 1, lines 10-25).

22. Claims 12 and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) in view of Narayanaswami (US005973705A).

Davis is relied upon for the teachings as discussed above relative to Claim 9.

However, Davis does not teach address space of data memory has frame buffer unit and texture memory unit. But, Narayanaswami teaches SIMD graphics processing system having frame buffer unit (frame buffer 110f, Fig. 2A) while implicitly suggesting texture memory unit.

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis so address space of data memory has frame buffer unit and texture memory unit because Narayanaswami teaches it reduces processing time (col. 2, lines 20-22).

23. Claims 14, 16, 17, and 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) in view of Krishna (US006161173A).

24. As per Claim 14, Davis teaches a method of executing instructions from a plurality of programs comprising: identifying N programs of said plurality of programs wherein each of the plurality of programs comprises a plurality of instructions (col. 2, lines 15-18); interleaving instructions from said N programs in a processor pipeline (col. 4, lines 13-15; col. 5, lines 52-54) wherein said pipeline has an average latency of one instruction per cycle (col. 1, lines 28-30; col. 7, lines 30-34) and wherein said pipeline has a depth of a plurality of execution stages (three execution stages) (col. 2, lines 12-28); and wherein an instruction from another of said N programs has been interleaved while said first instruction is executing (col. 2, lines 31-39; col. 4, lines 13-15; col. 5, lines 52-54).

However, Davis does not teach executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of

said one of said N programs wherein no no-op or idle is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction. However, Applicant's disclosure describes inserting no-ops into instruction stream or retarding launching of new programs until first program finishes (p. 17, lines 8-11). So, when no no-op is inserted, that means first instruction is completed and execution of second instruction can begin. Krishna teaches local scheduling circuitry stops main scheduler from issuing selected operation if latency of another operation would create conflict with main scheduler issuing selected operation (col. 2, lines 56-60). So, it is ensured first instruction is completed before beginning execution of second instruction. So, next instruction is not provided to the pipeline until a previous instruction has completed. Operation is executed if no no-op is inserted into pipeline (col. 5, lines 36-38; col. 2, lines 41-45). So, when no no-op is inserted into pipeline, this ensures first instruction is completed before beginning execution of second instruction. So, when no-op is inserted, this means that operation is not ready to be executed. When associated operation which is to be executed is inserted, there is no no-op inserted, this means that associated operation is ready to be executed, meaning first instruction is completed and it is now okay to execute associated operation. So, no no-op or idle is inserted for purpose of ensuring first instruction is completed before beginning execution of second instruction.

It would have been obvious to one of ordinary skill in the art at the time of invention by applicant to modify Davis so next instruction from one of plurality of programs is not provided to pipeline until previous instruction of one of plurality of programs has completed because Krishna suggests sometimes latency of another instruction would create conflict with main scheduler issuing selected instruction, so in order to avoid this conflict, next instruction is not

provided to pipeline until previous instruction has completed (col. 2, lines 56-60). It would be obvious to modify Davis to include checking no no-op or idle is inserted into pipeline for ensuring next instruction is not provided to pipeline until previous instruction has completed because Krishna suggests no-op is needed for indicating previous instruction has not yet completed (col. 5, lines 26-38).

25. As per Claim 16, Davis teaches further including the step of assigning a register to each of said N programs (col. 2, lines 25-28).

26. As per Claim 17, Davis teaches wherein said processor pipeline has a depth of N (three) (col. 2, lines 12-28).

27. As per Claim 23, Davis teaches a programmable processor for executing a plurality of programs, said programmable processor comprising: an execution pipeline having an average pipeline latency of one instruction per cycle (col. 1, lines 28-30; col. 7, lines 30-34) and having a depth of a plurality of execution stages (three execution stages) (col. 2, lines 12-28) and a depth (three execution stages) less than or equal to the plurality of programs (three or more programs) (col. 4, lines 36-39) wherein each of the plurality of programs comprises a plurality of instructions (col. 2, lines 15-18); and an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution (col. 4, lines 13-15; col. 5, lines 52-54); and wherein an instruction from another of said N programs has been interleaved while said first instruction is executing (col. 2, lines 31-39; col. 4, lines 13-15; col. 5, lines 52-54).

However, Davis does not teach wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality

of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed. However, Applicant's disclosure describes inserting no-ops into instruction stream or retarding launching of new programs until first program finishes (p. 17, lines 8-11). So, when no no-op is inserted, that means first instruction is completed and execution of second instruction can begin. Krishna teaches local scheduling circuitry stops main scheduler from issuing selected operation if latency of another operation would create conflict with main scheduler issuing selected operation (col. 2, lines 56-60). So, it is ensured first instruction is completed before beginning execution of second instruction. So, next instruction is not provided to the pipeline until a previous instruction has completed. Operation is executed if no no-op is inserted into pipeline (col. 5, lines 36-38; col. 2, lines 41-45). So, when no no-op is inserted into pipeline, this ensures first instruction is completed before beginning execution of second instruction. So, when no-op is inserted, this means that operation is not ready to be executed. When associated operation which is to be executed is inserted, there is no no-op inserted, this means that associated operation is ready to be executed, meaning first instruction is completed and it is now okay to execute associated operation. So, no no-op is inserted for purpose of ensuring first instruction is completed before beginning execution of second instruction. This would be obvious for the same reasons given in the rejection for Claim 14.

28. Claim 15 is rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) and Krishna (US006161173A) in view of Eickemeyer (US006061710A).

Davis and Krishna are relied upon for teachings as discussed above relative to Claim 14.

However, Davis and Krishna do not teach further including the step of assigning a program counter to each of said N programs. However, Eickemeyer teaches further including the step of assigning a program counter to each of said N programs (col. 8, lines 57-60). This would be obvious for the same reasons given in the rejection for Claim 6.

29. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Davis (US005357617A) and Krishna (US006161173A) in view of Nguyen (US005961628A).

Claim 18 is similar in scope to Claim 8, and so is rejected under the same rationale.

30. Claim 33 is rejected under 35 U.S.C. 103(a) as being unpatentable over Joffe (US006330584B1) in view of Krishna (US006161173A).

Joffe teaches a method, by a programmable processor, of executing instructions from plurality of programs (col. 2, lines 66-67), assigning 1st output register slot to first of plurality of programs wherein each of the plurality of programs has plurality of instructions (col. 1, line 62-col. 2, line 11). If wait signal is asserted, instruction execution is not completed, so instruction will be executed again until wait signals are deasserted, then next instruction can be executed (col. 10, lines 20-24, 31-32), and process repeats until all instructions have been executed. Joffe teaches if task attempts to access unavailable resource, task is suspended. When resource becomes available, suspended task is resumed, and instruction accessing resource is re-executed. Task does not get access to same resource until after every other task sharing resource has finished accessing resource (col. 2, lines 29-39). If Wait signal is asserted, instruction execution is not completed and PC register is frozen, but task remains active, and instruction will be executed again starting next clock cycle. If Suspend/Wait signals are deasserted, the PC register is changed to point to next instruction (col. 10, lines 19-32). So, Joffe teaches checking to see if

Suspend/Wait signals are desasserted, which indicates instruction execution is completed (col. 10, lines 19-32) and resource has become available (col. 2, lines 32-34), and resource becomes available after every other task sharing resource has finished accessing resource (col. 2, lines 29-39). So, Joffe teaches checking to see if all of the programs are completed. So, Joffe teaches executing instructions of first program until program is completed; loading output of first program into its reserved space when first program is completed (col. 9, lines 26-41); checking to see if all of plurality of programs are completed (col. 2, lines 35-39). Wait signal is asserted if register is not available, and wait signal is deasserted if register is available for new instruction (col. 10, lines 20-24, 31-32). Each task (program) has separate register and separate flags (col. 2, lines 11-13). So, second output register slot is assigned to second program. If task attempts to access unavailable resource, task is suspended. When resource becomes available, suspended task is resumed, and instruction accessing resource is executed (col. 2, lines 29-34). So, Joffe teaches checking to see if second register slot is available to assign to second program from plurality of programs when first program is completed; checking to see if one or more instructions are available when at least one of the programs is not completed (col. 2, lines 35-39).

However, Joffe does not teach placing no-op when no more instructions are available. However, Krishna teaches information in each entry describes either no-op or associated operation which is to be executed (col. 5, lines 36-38). So, when there is no-op, that means that no more instructions are available. This would be obvious for reasons for Claim 14.

Allowable Subject Matter

31. Claims 25-31 are allowed, for reasons given in the Office Action dated March 28, 2007.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JONI HSU whose telephone number is (571)272-7785. The examiner can normally be reached on M-F 8am-5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kee Tung can be reached on 571-272-7794. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

JH

/Joni Hsu/
Examiner, Art Unit 2628